

RRRRRRRRRRRR		MMM		MMM	SSSSSSSSSSSS
RRRRRRRRRRRR		MMM		MMM	SSSSSSSSSSSS
RRRRRRRRRRRR		MMM		MMM	SSSSSSSSSSSS
RRR	RRR	MMMMMM	MMMMMM	SSS	
RRR	RRR	MMMMMM	MMMMMM	SSS	
RRR	RRR	MMMMMM	MMMMMM	SSS	
RRR	RRR	MMM	MMM	SSS	
RRR	RRR	MMM	MMM	SSS	
RRR	RRR	MMM	MMM	SSS	
RRRRRRRRRRRR		MMM		SSSSSSSSSS	
RRRRRRRRRRRR		MMM		SSSSSSSSSS	
RRRRRRRRRRRR		MMM		SSSSSSSSSS	
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM		SSSSSSSSSSSS	
RRR	RRR	MMM		SSSSSSSSSSSS	
RRR	RRR	MMM		SSSSSSSSSSSS	

\_S

Syn

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

PI

```

RRRRRRRR      MM      MM      000000      CCCCCCCC      000000      MM      MM      CCCCCCCC      LL      NN      NN
RRRRRRRR      MM      MM      000000      CCCCCCCC      000000      MM      MM      CCCCCCCC      LL      NN      NN
RR      RR      MMMM      MMMM      00      00      CC      00      00      MMMM      MMMM      CC      LL      NN      NN
RR      RR      MMMM      MMMM      00      00      CC      00      00      MMMM      MMMM      CC      LL      NN      NN
RR      RR      MM      MM      MM      00      0000      CC      00      00      MM      MM      CC      LL      NNNN      NN
RR      RR      MM      MM      MM      00      0000      CC      00      00      MM      MM      CC      LL      NNNN      NN
RRRRRRRR      MM      MM      00      00      00      CC      00      00      MM      MM      CC      LL      NN      NN
RRRRRRRR      MM      MM      00      00      00      CC      00      00      MM      MM      CC      LL      NN      NN
RR      RR      MM      MM      MM      0000      00      CC      00      00      MM      MM      CC      LL      NN      NN
RR      RR      MM      MM      MM      0000      00      CC      00      00      MM      MM      CC      LL      NN      NN
RR      RR      MM      MM      MM      00      00      CC      00      00      MM      MM      CC      LL      NN      NN
RR      RR      MM      MM      MM      00      00      CC      00      00      MM      MM      CC      LL      NN      NN
RR      RR      MM      MM      MM      000000      000000      MM      MM      CCCCCCCC      LLLLLLLLLL      NN      NN
RR      RR      MM      MM      MM      000000      000000      MM      MM      CCCCCCCC      LLLLLLLLLL      NN      NN

```

```

LL      I I I I I      SSSSSSSS
LL      I I I I I      SSSSSSSS
LL      I I          SS
LL      I I          SS
LL      I I          SS
LL      I I          SS
LL      I I          SSSSSS
LL      I I          SSSSSS
LL      I I          SS
LL      I I          SS
LL      I I          SS
LL      I I          SS
LLLLLLLLLL      I I I I I      SSSSSSSS
LLLLLLLLLL      I I I I I      SSSSSSSS

```

(2)	71
(3)	105

DECLARATIONS  
RMSDISCOMMON - COMMON CLEANUP ON CONNECT-DISCONNECT ROUTINES

RMSY  
\$\$  
\$\$  
\$\$\$  
\$\$\$  
\$\$\$  
\$\$\$  
BI  
BI  
BLI  
BLI  
BLI  
CHI  
CSI  
CSI  
DE  
EX  
FA  
FT  
GB  
GB  
GB  
GB  
IFI  
IFI  
IFI  
IFI  
IFI  
IFI  
IFI  
IFI  
IMI  
IMI  
IMI  
IMI  
INI  
IRI  
IRI  
IRI  
IRI  
IRI  
IRI  
IRI  
IRI  
IRI  
IRI  
NON  
NTS  
NTP  
PIG  
PIG  
PIG  
RAI  
RL



```

0000 1      $BEGIN RMOCOMCLN,000,RM$RMS0,<COMMON CLEAN UP CONN-DISCONN>
0000 2
0000 3
0000 4 *****
0000 5 *****
0000 6 *****
0000 7 *****
0000 8 *****
0000 9 *****
0000 10 *****
0000 11 *****
0000 12 *****
0000 13 *****
0000 14 *****
0000 15 *****
0000 16 *****
0000 17 *****
0000 18 *****
0000 19 *****
0000 20 *****
0000 21 *****
0000 22 *****
0000 23 *****
0000 24 *****
0000 25 *****
0000 26 *****
0000 27 *****
0000 28 *****
0000 29 *****
0000 30 *****
0000 31 *****
0000 32 *****
0000 33 *****
0000 34 *****
0000 35 *****
0000 36 *****
0000 37 *****
0000 38 *****
0000 39 *****
0000 40 *****
0000 41 *****
0000 42 *****
0000 43 *****
0000 44 *****
0000 45 *****
0000 46 *****
0000 47 *****
0000 48 *****
0000 49 *****
0000 50 *****
0000 51 *****
0000 52 *****
0000 53 *****
0000 54 *****
0000 55 *****
0000 56 *****
0000 57 *****

      $COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
      $DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
      $ALL RIGHTS RESERVED.

      $THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
      $ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
      $INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
      $COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
      $OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
      $TRANSFERRED.

      $THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
      $AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
      $CORPORATION.

      $DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
      $SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

      *****

      ++
      Facility: rms32

      Abstract:
      this module provides four entry points to provide common
      clean up on connect - disconnect.

      Environment:
      star processor running starlet exec.

      Author: L F Laverdure,          Creation Date: 31-MAR-1977

      Modified By:

      V03-008 JWT0160          Jim Teague          29-Feb-1984
      Complete the anticipated removal of RM$DEALLEFN.

      V03-007 SHZ0001          Stephen H. Zalewski    3-Feb-1983
      Make routine RM$DEALLEFN a NOP in anticipation of
      removing it altogether.

      V03-006 KPL0008          Peter Lieberwirth      29-Apr-1983
      Always deallocate IRAB journaling structures.

      V03-005 KBT0367          Keith B. Thompson      11-Oct-1982
      Deallocate irab and asb seperatly

      V03-004 JWH0002          Jeffrey W. Horn        20-Sep-1982
      Rename RLB$W_OWNER to RLB$L_OWNER.

      V03-003 KBT0322          Keith B. Thompson      9-Sep-1982

```

0000 58 :  
0000 59 :  
0000 60 :  
0000 61 :  
0000 62 :  
0000 63 :  
0000 64 :  
0000 65 :  
0000 66 :  
0000 67 :--  
0000 68 :  
0000 69 :

Remove all S0 sharing code

- V03-002 KBT0202 Keith B. Thompson 23-Aug-1982  
Reorganize psects and fix rev. history of jwh0001
- V03-001 JWH0001 Jeffrey W. Horn 19-May-1982  
Add call to RMSDSCJNL to get rid of journaling BDB and  
buffer.

```

0000 71      .SBTTL DECLARATIONS
0000 72
0000 73 :
0000 74 : Include Files:
0000 75 :
0000 76 :
0000 77 :
0000 78 : Macros:
0000 79 :
0000 80
0000 81      $BLBDEF
0000 82      $GBPBDEF      ; global buffer pointer block
0000 83      $IFBDEF
0000 84      $DEVDEF
0000 85      $IRBDEF
0000 86      $CSHDEF
0000 87      $RLSDEF
0000 88      $RLBDEF
0000 89      $FABDEF
0000 90      $RABDEF
0000 91      $PIODEF
0000 92      $IMPDEF
0000 93      $PSLDEF
0000 94      $RMSDEF
0000 95
0000 96 :
0000 97 : Equated Symbols:
0000 98 :
0000 99 :
0000 100 :
0000 101 : Own Storage:
0000 102 :
0000 103

```



```

0000 105      .SBTTL  RM$DISCOMMON - COMMON CLEANUP ON CONNECT-DISCONNECT ROUTINES
0000 106
0000 107      :++
0000 108      : RM$DISCOMMON
0000 109
0000 110
0000 111      RM$DISCOMMONSUC - sets r0 to rm$_suc and falls thru to rm$discommon
0000 112      RM$DISCOMMON    - checks for network disconnect and falls thru to
0000 113                      rm$comclnup
0000 114      RM$COMCLNUP    - return all bdb's and buffers for stream and fall thru
0000 115                      to rm$_cln1 r0 already pushed onto stack
0000 116      RM$_CLN1      - deallocate irab and zero isi and irab table entry
0000 117
0000 118      Calling sequence:
0000 119
0000 120      bsbw  rm$discommonsuc
0000 121      bsbw  rm$discommon
0000 122      brw   rm$comclnup
0000 123      bsbw  rm$_cln1
0000 124
0000 125      Input Parameters:
0000 126
0000 127      r11    impure area address
0000 128      r10    ifab address
0000 129      r9     irab address
0000 130      r8     rab address
0000 131      r0     status code (except for entry at rm$discommonsuc)
0000 132
0000 133      Implicit Inputs:
0000 134
0000 135      for entry at rm$discommon:  irb$_vap_conn
0000 136      for entry at rm$comclnup:  irb$_bcnt & bdb chain
0000 137      for entry at rm$_cln1:    irb$_irab_lnk
0000 138
0000 139      Output Parameters:
0000 140
0000 141      r0      status code
0000 142      r7      set from irb$_mode
0000 143      r9      zeroed
0000 144      ap      set from irb$_arglst
0000 145      r1-r6   destroyed
0000 146
0000 147      Implicit Outputs:
0000 148
0000 149      rab$_isi zeroed
0000 150      rab$_stv possibly updated
0000 151      irab, its bcb's, bdb's and related buffers deallocated
0000 152
0000 153      Completion Codes:
0000 154
0000 155      standard rms. if an error occurs it will replace
0000 156      the status code input in r0, otherwise the code input
0000 157      in r0 will be used.
0000 158
0000 159      Side Effects:
0000 160
0000 161      none

```

RMOCOMCLN  
V04-000

COMMON CLEAN UP CONN-DISCONN  
RMSDISCOMMON - COMMON CLEANUP ON CONNECT

G 2

16-SEP-1984 00:14:09  
5-SEP-1984 16:21:29

VAX/VMS Macro V04-00  
[RMS.SRC]RMOCOMCLN.MAR;1

Page 5  
(3)

0000 162 :--  
0000 163



```

0000 165
0000 166 :++
0000 167 :
0000 168 : entry point to set r0 = rm$_suc and fall thru into rm$discommon
0000 169 :
0000 170 :--
0000 171 :
0000 172 RM$DISCOMMONSUC::
0000 173     RMSSUC
0003 174 :
0003 175 :++
0003 176 :
0003 177 : entry point to check for network disconnect and fall thru into rm$comclnup
0003 178 :
0003 179 :--
0003 180 :
0003 181 RM$DISCOMMON::
50 DD 0003 182     PUSHL    R0                ; save status
0005 183 :
0005 184 :
0005 185 :     check for process-permanent file cleanup
0005 186 :
0005 187 :
69 22 E1 0005 188     BBC      #IRB$V_PPF_IMAGE,(R9),-
03 03 0008 189     1$                ; branch if not indirect acc to ppf
00DD 31 0009 190     BRW      IND FILE                ; branch
6A 2E E1 000C 191 1$: BBC      #IFB$V_PPF_INPUT,(R10),-
10 10 000F 192     10$                ; branch if not sys$input
2401 8F B0 0010 193     MOVW     #1+<^A/$/28>,-
00000000'9F 0014 194     @#PIO$GT_ENDSTR                ; reset eod string to '$'
02 8A 0019 195     BICB2    #1@PIO$V_EOD,-
00000000'9F 001B 196     @#PIO$GW_STATUS                ; and clear eod flag
69 3C E5 0020 197 10$: BBCC    #IRB$V_DAP_CONN,(R9),-
0D 0D 0023 198     BIOCHK    ; not net then continue
0024 200 :
0024 201 :++
0024 202 :
0024 203 :     perform network disconnect function
0024 204 :
0024 205 :--
0024 206 :
0024 207 NTDISC:
6B 04 E0 0024 208     BBS      #IMB$V_IORUNDOWN,(R11),-
09 09 0027 209     BIOCHK    ; branch if i/o rundown in progress
FFD5' 30 0028 210     BSBW     NT$DISCONNECT                ; disconnect at remote node
03 50 E8 002B 211     BLBS     R0,BIOCHK                ; branch if successful
6E 50 D0 002E 212     MOVL     R0,(SP)                ; save error code
0031 213 :
0031 214 BIOCHK:
0B 22 E0 0031 215     BBS      #IFB$V_BIO,-
23 AA 95 0033 216     IFB$B_FAC(R10),NOBUFF                ; branch if block i/o
0C 13 0036 217     TSTB     IFB$B_ORGCASE(R10)                ; is this seq f.o. ?
6A 33 E0 0039 218     BEQL     RM$COMCLNUP                ; if yes, no lock bdb to return
08 08 003B 219     BBS      #IFB$V_NORECLK,(R10),-
2A 11 003E 220     RM$COMCLNUP                ; branch if no locking
003F 221     BRB      RTNBLB                ; return the lock blb.

```

```

0041 222 NOBUFF:
0041 223
0041 224 :
0041 225 : This is the return the block i/o bdb. This will branch into cache
0041 226 : and release to get rid of it. Because the bdb was not counted as
0041 227 : a buffer when allocated, the avlcl count is bumped so cache will
0041 228 : just find it and take it. Cache will also set the bdb$w_users count
0041 229 : to 1 so that release is not upset for the relative and isam orgs.
0041 230 : The only reason cache and release are used at all is because if
0041 231 : someone is out there doing asynch multistreaming with block i/o,
0041 232 : this should prevent us from returning a bdb in use, because cache
0041 233 : will look for one with a users count of 0, and the block i/o code
0041 234 : sets the users count to 1 when using it.
0041 235 :
0041 236 :
0084 CA B6 0041 237 INCW IFB$W_AVLCL(R10) ; want to fake out cache so it
0045 238 ; doesn't try to free one.
05 11 0045 239 BRB BIO ; branch to return it.
0047 240
0047 241 :++
0047 242 :
0047 243 : entry point to return all blb's, bdb's and buffers for this stream.
0047 244 : status already pushed onto stack. this is error path from rm$bdballoc
0047 245 : to return bdb's, blb's allocated before failure. irab will also be
0047 246 : deallocated before returning to user so no other structures can be present.
0047 247 : this is strictly error path on connect operation. lock blb will not have
0047 248 : been allocated.
0047 249 :
0047 250 :--
0047 251 :
0047 252 RMSCOMCLNUP::
0047 253
0047 254 :++
0047 255 :
0047 256 : return bdb's used by this stream
0047 257 :
0047 258 :--
0047 259 :
54 A9 97 0047 260 DECB IRB$B_BCNT(R9) ; decrement buffer count
48 19 004A 261 BLSS CHKGBL ; branch if no more
004C 262 BIO: $CACHE VBN=#0,SIZE=#0,-
004C 263 ; get any BDB.
EE 50 E9 0056 264 BLBC RO,RMSCOMCLNUP ; failed so go around again
53 01 D0 0059 265 RTNBDB: MOVL #RL$M_RETURN,R3 ; set return flag
FFA1' 30 005C 266 BSBW RMSRELEASE ; release bdb & buffer
03 50 E8 005F 267 BLBS RO,RTNJNL ; go check for blb release
6E 50 D0 0062 268 MOVL RO,(SP) ; save error code
0065 269
00000000'EF 16 0065 270 RTNJNL: JSB RMSDSCJNL ; clean up IRAB journal structures
006B 271
006B 272 :++
006B 273 :
006B 274 : check for locking and return blb's if so.
006B 275 :
006B 276 :--
006B 277 :
006B 278 RTNBLB:

```

```

DB 6A 33 E0 006B 279 BBS #IFBSV_NORECLK, (R10), RMSCOMCLNUP ; branch back if no locking.
      02 10 006F 280 BSBB RTNBLBS ; Return a BLB.
      D4 11 0071 281 BRB RMSCOMCLNUP ; Loop to get any more.
      0073 282 RTNBLBS:
      0073 283 ASSUME BLBSL_FLNK EQ 0
      0073 284 ASSUME BLBSL_BLNK EQ 4
54 0098 CA DE 0073 285 MOVAL IFBSL_BLBFLNK(R10), R4 ; get list head.
      50 54 D0 0078 286 MOVL R4, R0 ; save for end test.
54 04 A4 D0 007B 287 10$: MOVL 4(R4), R4 ; get blb element.
      54 50 D1 007F 288 CMPL R0, R4 ; back at list head?
      09 13 0082 289 BEQL 20$ ; it's a bug if we are.
      24 A4 D5 0084 290 TSTL BLBSL_LOCK_ID(R4) ; is this one in use?
      F2 12 0087 291 BNEQ 10$ ; NEQ it is, get another.
      FF74 31 0089 292 BRW RMSRETLB ; and return it.
      05 008C 293 RSB ; Return.
      008D 294 20$:
      008D 295 RMSPPBUG FTL$_NOBLB
      0094 296
      0094 297 CHKGBL:
27 69 36 E1 0094 298 BBC #IRBSV_GBLBUFF, (R9), RTNRLB ; Branch if no gbp, blb allocated.
      7E D4 0098 299 CLRL -(SP) ; Init pass counter.
54 40 AA DE 009A 300 10$: MOVAL IFBSL_BDB_FLNK(R10), R4 ; Get list head address.
      50 54 D0 009E 301 MOVL R4, R0 ; Save for end test.
      00A1 302
      00A1 303 ASSUME IFBSL_BDB_BLNK EQ <IFBSL_BDB_FLNK + 4>
      00A1 304 ASSUME GBPBSL_BLNK EQ 4
      00A1 305
54 04 A4 D0 00A1 306 20$: MOVL 4(R4), R4 ; Scan backwards.
      50 54 D1 00A5 307 CMPL R4, R0 ; Back at head?
      12 13 00A8 308 BEQL 30$ ; Continue if so.
      00AA 309 ASSUME <GBPBSL_BID & 1> EQ 1
      F3 08 A4 E9 00AA 310 BLBC GBPBSL_BID(R4), 20$ ; Keep looking if not GBP.
      0C A4 B5 00AE 311 TSTW GBPBSL_USERS(R4) ; Is use count zero?
      EE 12 00B1 312 BNEQ 20$ ; Keep looking if not.
      FF4A 30 00B3 313 BSBW RMSRETLB ; Return the GBP.
      BB 10 00B6 314 BSBB RTNBLBS ; Return the BLB.
DE 6E 00 E3 00B8 315 BBCS #0, (SP), 10$ ; Branch if 1st pass.
      5E 04 C0 00BC 316 30$: ADDL2 #4, SP ; Remove pass counter.
      00BF 317
      00BF 318 ;++
      00BF 319
      00BF 320 ; unlock all locked records for this stream and deallocate all unused rlb's
      00BF 321
      00BF 322 ;--
      00BF 323
      00BF 324 RTNRLB: BSBW RMSUNLOCKALL
53 59 38 C1 00C2 325 ADDL3 #IRBSL_RLB_LNK, R9, R3 ; get rlb list head addr in r3
      5C 53 D0 00C6 326 MOVL R3, AP ; copy it
      00C9 327 ASSUME RLB$S_LNK EQ 0
      54 6C D0 00C9 328 40$: MOVL (AP), R4 ; get next rlb addr
      19 13 00CC 329 BEQL 60$ ; branch if no more
      10 A4 D5 00CE 330 TSTL RLB$S_OWNER(R4) ; in use?
      0F 12 00D1 331 BNEQ 55$ ; branch if yes
      6C 64 D0 00D3 332 MOVL RLB$S_LNK(R4), (AP) ; remove rlb from chain
      52 1C D0 00D6 333 MOVL #RLBSL_BLN, R2 ; set rlb length
      53 DD 00D9 334 PUSHL R3 ; save space header addr
      FF22 30 00DB 335 BSBW RMSRETLB ; deallocate rlb

```



RM0COMCLN  
V04-000

COMMON CLEAN UP CONN-DISCONN  
RMSDISCOMMON - COMMON CLEANUP ON CONNECT

K 2

16-SEP-1984 00:14:09 VAX/VMS Macro V04-00  
5-SEP-1984 16:21:29 [RMS.SRC]RM0COMCLN.MAR;1

Page 9  
(5)

SC	08	BA	00DE	336	POPR	#^M<R3>	; restore space header addr
	E7	11	00E0	337	BRB	40\$	; go get next rlb
	54	D0	00E2	338 55\$:	MOVL	R4,AP	; rlb in use - copy addr
	E2	11	00E5	339	BRB	40\$	; go get next rlb
	0F	11	00E7	340 60\$:	BRB	RTNIRB	

```

00E9 342
00E9 343 :++
00E9 344 :
00E9 345 : this is a disconnect of an indirectly connected irab (i.e., for the image)
00E9 346 :
00E9 347 : just zero isi and clear eof if unit record device
00E9 348 :
00E9 349 :--
00E9 350
00E9 351 IND_FILE:
02 A8 B4 00E9 352 CLRW RAB$W_ISI(R8) ; clear isi
00 00 E1 00EC 353 BBC #DEV$? REC,-
41 6A 00EE 354 IFB$L PRIM_DEV(R10),EXIT; branch if not unit record device
00F0 355 CSB #IRB$?_EOF,(R9) ; reset eof flag
3B 11 00F4 356 BRB EXIT

```

```

00F6 358
00F6 359 :++
00F6 360 :
00F6 361 : entry point for when no bdb's or buffers allocated.
00F6 362 : simply deallocate the irab and zeroes isi.
00F6 363 :
00F6 364 :--
00F6 365
00F6 366 RMSCCLN1::
50 DD 00F6 367 PUSHL R0 ; save error code
00F6 368 ; find this irab in irab chain
00F6 369
53 SA D0 00F6 370 RTNIRB: MOVL R10,R3 ; get irab addr
00F6 371 ASSUME IRB$_IRAB_LNK EQ IFB$_IRAB_LNK
56 1C A3 D0 00F6 372 10$: MOVL IRB$_IRAB_LNK(R3),R6 ; get next irab
59 56 D1 00F6 373 CMPL R6,R9 ; is this the one?
05 13 0102 374 BEQL 20$ ; branch if yes
53 56 D0 0104 375 MOVL R6,R3 ; move ptr to other reg
F2 11 0107 376 BRB 10$ ; & keep searching
0109 377
0109 378 :
0109 379 : got the irab - close up chain and deallocate the irab
0109 380 :
0109 381
1C A3 1C A6 D0 0109 382 20$: MOVL IRB$_IRAB_LNK(R6),IRB$_IRAB_LNK(R3)
010E 383
010E 384 :
010E 385 : restore the user's mode and arg list pointer from the irab before
010E 386 : deallocating it.
010E 387 :
010E 388
57 0A A9 9A 010E 389 MOVZBL IRB$_MODE(R9),R7
5C 18 A9 D0 0112 390 MOVL IRB$_ARGLST(R9),AP
0116 391
0116 392 :
0116 393 : deallocate asb and irab
0116 394 :
0116 395
53 5A D0 0116 396 MOVL R10,R3 ; get space header page
54 14 A9 D0 0119 397 MOVL IRB$_ASBADDR(R9),R4 ; get asb addr
06 13 011D 398 BEQL 30$ ; just in case we don't have one?
FEDE' 30 011F 399 BSBW RMSRETLK ; deallocate irab
53 5A D0 0122 400 MOVL R10,R3 ; restore header page
54 59 D0 0125 401 30$: MOVL R9,R4 ; get irab addr
FED5' 30 0128 402 BSBW RMSRETLK ; deallocate irab
51 1C AB D0 012B 403 MOVL IMP$_IRABTBL(R11),R1 ; get irab table addr
07 10 012F 404 BSBB ZAPCOM ; zero isi & isi table entry
01 BA 0131 405 EXIT: POPR #^M<R0> ; restore status
05 0133 406 RSB ; & return

```



```

0134 408
0134 409 :++
0134 410
0134 411 : subroutine to clear the ifab or irab table
0134 412 : entry for the ifab or irab whose address is in r9.
0134 413 : also zeros the ifi or isi and r9
0134 414
0134 415 : inputs:
0134 416 : r11 impure area address
0134 417 : r9 ifab/irab address
0134 418 : r8 fab/rab address
0134 419
0134 420 : outputs:
0134 421 : ifab/irab table pointer zeroed.
0134 422 : ifab/irab address in r9 zeroed.
0134 423 : fab$w_ifi/rab$w_isi zeroed
0134 424 : r0-r2 destroyed
0134 425
0134 426 :--
0134 427
0134 428 RMSZAPIFI::
51 18 AB DE 0134 429 MOVAL IMP$L_IFABTBL(R11),R1 ; get ifab table addr
0138 430 ZAPCOM:
52 04 A1 DE 0138 431 MOVAL 4(R1),R2 ; leave r1 pointing to link
013C 432 ; point r2 to 1st entry
013C 433
50 20 AB 3C 013C 434 MOVZWL IMP$W_ENTPERSEG(R11),R0 ; # entries per table segment.
59 82 D1 0140 435 10$: CMPL (R2)+,R9 ; is this desired entry?
08 13 0143 436 BEQL 20$ ; branch if yes
F8 50 F5 0145 437 SOBGTR R0,10$ ; keep trying
51 61 D0 0148 438 MOVL (R1),R1 ; next segment
EB 11 014B 439 BRB ZAPCOM
014D 440
014D 441 : this is the sought-for table entry - zero it
014D 442 :
014D 443 :
014D 444
72 D4 014D 445 20$: CLRL -(R2)
014F 446 ASSUME RAB$W_ISI EQ FAB$W_IFI
02 A8 B4 014F 447 CLRW RAB$W_ISI(R8) ; zero isi (or ifi)
22 AB B7 0152 448 DECW IMP$W_NUM_IFABS(R11) ; decrement count of allocated ifabs
59 D4 0155 449 CLRL R9 ; zero ifab or irab address
05 0157 450 RSB
0158 451
0158 452 .END

```

RMOCOMCLN  
Symbol table

COMMON CLEAN UP CONN-DISCONN

B 3

16-SEP-1984 00:14:09 VAX/VMS Macro V04-00  
5-SEP-1984 16:21:29 [RMS.SRC]RMOCOMCLN.MAR;1

Page 13  
(11)

```

$$PSECT_EP      = 00000000
$$TMP           = 00000004
$$RMSTEST       = 0000001A
$$RMS_PBUGCHK   = 00000010
$$RMS_TBUGCHK   = 00000008
$$RMS_UMODE     = 00000004
BIO             = 0000004C R    01
BIOCHK          = 00000031 R    01
BLBSL_BLNK      = 00000004
BLBSL_FLNK      = 00000000
BLBSL_LOCK_ID   = 00000024
CHKGBL         = 00000094 R    01
CSHSM_NOBUFFER  = 00000008
CSHSM_NOREAD    = 00000004
DEVSV_REC       = 00000000
EXIT            = 00000131 R    01
FABSW_IFI       = 00000002
FTLS_NOBLB     = 00000000
GBPBSB_BID      = 00000008
GBPBSB_BID      = 00000015
GBPBSB_BLINK    = 00000004
GBPBSB_USERS    = 0000000C
IFBSB_FAC       = 00000022
IFBSB_ORGCASE   = 00000023
IFBSL_BDB_BLNK  = 00000044
IFBSL_BDB_FLNK  = 00000040
IFBSL_BLBFLNK   = 00000098
IFBSL_IRAB_LNK  = 0000001C
IFBSL_PRIM_DEV  = 00000000
IFBSV_BIO       = 00000005
IFBSV_NORECLK   = 00000033
IFBSV_PPF_INPUT = 0000002E
IFBSW_AVLCL     = 00000084
IMPSL_IFABTBL   = 00000018
IMPSL_IRABTBL   = 0000001C
IMPSV_IORUNDOWN = 00000004
IMPSW_ENTPERSEG = 00000020
IMPSW_NUM_IFABS = 00000022
IND_FILE        = 000000E9 R    01
IRBSB_BCNT      = 00000054
IRBSB_MODE      = 0000000A
IRBSL_ARGLST    = 00000018
IRBSL_ASBADDR   = 00000014
IRBSL_IRAB_LNK  = 0000001C
IRBSL_RLB_LNK   = 00000038
IRBSV_DAP_CONN  = 0000003C
IRBSV_EOF       = 00000021
IRBSV_GBLBUFF   = 00000036
IRBSV_PPF_IMAGE = 00000022
NOBUFF         = 00000041 R    01
NT$DISCONNECT   = ***** X    01
NTDISC         = 00000024 R    01
PIO$GT_ENDSTR   = ***** X    01
PIO$GW_STATUS   = ***** X    01
PIO$V_EOD       = 00000001
RABSW_ISI       = 00000002
RLB$C_BLN       = 0000001C

```

```

RLB$C_LNK       = 00000000
RLB$C_OWNER     = 00000010
RLS$M_RETURN    = 00000001
RMSBUG          = ***** X    01
RMS$CACHE       = ***** X    01
RMS$CCLN1       = 000000F6 RG    01
RMS$COMCLNUP    = 00000047 RG    01
RMS$DISCOMMON   = 00000003 RG    01
RMS$DISCOMMONSUC = 00000000 RG    01
RMS$DSCJNL      = ***** X    01
RMS$RELEASE     = ***** X    01
RMS$RETLB       = ***** X    01
RMS$RETLBK      = ***** X    01
RMS$RETLBPB     = ***** X    01
RMS$RETSPC1     = ***** X    01
RMS$UNLOCKALL   = ***** X    01
RMS$ZAPIFI      = 00000134 RG    01
RTNBDB          = 00000059 R    01
RTNBLB          = 0000006B R    01
RTNBLBS         = 00000073 R    01
RTNIRB          = 000000F8 R    01
RTNJNL          = 00000065 R    01
RTNRLB          = 000000BF R    01
ZAPCOM          = 00000138 R    01

```



-----  
! Psect synopsis !  
-----

PSECT name	Allocation	PSECT No.	Attributes
ABS	00000000 ( 0.)	00 ( 0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
RM\$RMSO	00000158 ( 344.)	01 ( 1.)	PIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC BYTE
\$ABSS	00000000 ( 0.)	02 ( 2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE

-----  
! Performance indicators !  
-----

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.07	00:00:00.89
Command processing	114	00:00:00.71	00:00:05.02
Pass 1	347	00:00:11.72	00:00:35.48
Symbol table sort	0	00:00:01.53	00:00:02.32
Pass 2	93	00:00:02.35	00:00:05.38
Symbol table output	11	00:00:00.13	00:00:01.72
Psect synopsis output	2	00:00:00.03	00:00:00.16
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	598	00:00:16.56	00:00:50.98

The working set limit was 1650 pages.  
65097 bytes (128 pages) of virtual memory were used to buffer the intermediate code.  
There were 60 pages of symbol table space allocated to hold 1211 non-local and 16 local symbols.  
452 source lines were read in Pass 1, producing 14 object records in Pass 2.  
31 pages of virtual memory were used to define 30 macros.

-----  
! Macro library statistics !  
-----

Macro library name	Macros defined
\$255\$DUA28:[RMS.OBJ]RMS.MLB;1	19
\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	1
\$255\$DUA28:[SYSLIB]STARLET.MLB;2	6
TOTALS (all libraries)	26

1366 GETS were required to define 26 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LISS:RMOCOMCLN/OBJ=OBJ\$:RMOCOMCLN MSRC\$:RMOCOMCLN/UPDATE=(ENH\$:RMOCOMCLN)+EXECML\$/LIB+LIB\$:RMS/LIB



0318

AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY